Remarks

Claims 1-9 and 11-19 are pending in the application. Claims 1 and 13 have been amended herein. Favorable reconsideration of the application, as amended, is respectfully requested.

I. REJECTION OF CLAIMS 1-9 AND 11-19 UNDER 35 USC §112

A. Rejection of Claims 1-9 and 11- 19 under 35 USC §112, first paragraph

Claims 1-9 and 11-19 stand rejected under 35 USC §112, first paragraph, as failing to comply with the written description requirement. Withdrawal of the rejection is respectfully requested for at least the following reasons.

The Examiner states that the claims contain subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventors, at the time the application was filed, had possession of the claimed invention. Specifically, the Examiner asserts that the limitation "analyzing the loop termination condition to determine whether it is possibly non-terminating" is not described in the specification. Applicants respectfully disagree with the Examiner.

Initially, applicants wish to point out that the Examiner has misinterpreted applicants' arguments presented in the reply to the previous Office Action. In the previous reply, applicants argued that "possibly non-terminating" is different from "definitely non terminating" (which would be required to solve the halting problem). Contrary to the Examiner's contention, applicants did not argue that the portion of the specification at page 50, lines 13-14 was not what was claimed. Instead, applicants stated the "Examiner refers to page 50, lines 13-14 as describing a method that would include solving the halting problem" (emphasis added). Applicants then stated the Examiner's assertion is irrelevant, as the claims do not recite a method of solving the halting problem. Applicants further stated "whether or not the specification discloses a method that could be interpreted as solving the halting problem is not at issue" (emphasis added). Applicants respectfully request that the Examiner carefully review this argument of the previous reply so as to appreciate applicants' position.[1]

[1] See pages 8-9, section I(a) of the reply to the Office Action dated April 25, 2005, which was filed on August 25, 2005.

Regarding the Examiner's current rejection, the invention improves the speed of simulation, for example, by finding certain cases were an optimization can be applied. These cases are loops which are known to terminate.

The optimization lies in NOT inserting an exit point (because it is unnecessary and inefficient in computing time) into known terminating loops (whereas every other loop does receive an exit point). Therefore, the more loops in a program that can be identified as always terminating, the better an optimization the invention can deliver.

Loops are NOT optimized in this manner if it is unknown whether they will terminate, because this will alter the behavior of the simulation. That is, possibly terminating loops also receive an exit point. Thus, the test may err, so long as it errs on the side of saying a terminating loop is of unknown termination, and never the other way around.

As the Examiner rightly states, such loops cannot be categorized exactly, because that would entail solving the infamous halting problem. However, it is enough to identify some cases. The disclosure provides some simple methods (such as spotting certain patterns) to detect guaranteed termination. For example, and referring back to page 50, lines 1-14 of the application, there is discussed analyzing the loop to check whether it is "possibly non-terminating", or whether it "definitely terminates". The cited portion then describes methods for checking whether a loop <u>definitely terminates</u>, including: (a) checking if the loop represents a standard for loop; and (b) checking whether every execution of the S-body reaches an exit point. Any loop which doesn't match the pattern would then be simply classified as possibly non-terminating. Here "possibly" refers to the fact that the optimizer doesn't know the termination condition, not that the loop itself is indeterminate.

The invention of claim 1 simply uses the results of such a method of detection to inform an optimization. A person skilled in the art of programming and compiler technology could easily come up with a number of syntactic tests which could implement a suitable method. It is clear from the Examiner's examples that the he understands what kind of things make loops hard to analyze, such as extra assignments to the loop counter.

In view of the above, it is respectfully submitted that if one can ascertain that a loop <u>definitely terminates</u> based on the provided examples, logically it follows, and one skilled in the art certainly would appreciate, that a loop is <u>possibly non-terminating</u> if these conditions are not satisfied. In other words, since one cannot ascertain that the loop definitely terminates, it is possibly non-terminating.

With respect to the Examiner's examples, the first example (multiply by three and add one or divide by two loop) is a famous mathematical example of a loop which nobody knows whether or not it terminates, and clearly not a candidate for simple detection.

In the second example, this isn't a "standard for-loop", because the loop counter is incremented each time (i++), but it's also assigned elsewhere in the loop (i=0), so it doesn't match the description "the counter is increased/decreased monotonically". Therefore it is marked as "possibly non-terminating". There are programming languages which do not allow assignment to the loop counter within a for-loop for precisely this reason.

In the third example, the assignment i=i is clearly without effect, and it is plausible that it would be removed from consideration during early stages of compiling. If it is removed then it becomes a standard for-loop. If it isn't removed then again it isn't a standard for-loop (it appears to modify the loop counter), so it is "possibly non-terminating".

To further clarify the claimed invention, claim 1 has been amended to change "possibly non-terminating" to "not automatically determined to be definitely terminating".

In view of the above discussion, it is respectfully submitted that there is sufficient disclosure in the application to enable one skilled in the art to understand how to determine whether a loop is not automatically determined to be definitely terminating, as recited in claim 1.

Accordingly, withdrawal of the rejection is respectfully requested.

B. Rejection of claims 11 and 12 under 35 USC §112, second paragraph

Claims 11 and 12 stand rejected under 35 USC §112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicants regard as the invention. In particular, the Examiner asserts that it is unclear how the word "placing" is to be interpreted (i.e., whether "placing" means "replacing"). Withdrawal of the rejection is respectfully requested for at least the following reasons.

As shown in figures 15 through 17 of the application, and as well as page 34, line 11-page 38, line 13 of the specification, the scheduler:

(a) searches for an unhandled active process;
(b) chooses a current process from the list of unhandled active processes;
(c) selects an entry point for the current process;
(d) executes the current process until an exit point has been reached; and
(e) then places a new entry point at this exit point prior to repeating the steps (a)-(e).

Figure 15 illustrates that in the process handler unit 63, the current process begins an entry point and reaches an exit point, then execution returns to the scheduler 62, and the next process begins an entry point located at the exit point of the previous process (i.e., along the same line in figure 15), and execution of this present process occurs until another exit point is reached, etc. Figure 16 shows unhandled processes positioned side-by-side.

Taking these figures in the consideration along with the above cited portion of the specification, it is clear that the exit point of any one active process would not be "replaced" by a new entry point. Rather, a new entry point is placed at said exit point (or immediately following said exit point in a sequential code) such that the next unhandled active process (i.e., one that immediately follows the presently executed active process) can be accessed and executed by the scheduler as shown in the loop structures of figures 15 and 17.

In view of the above, one skilled in the art would understand the meaning of the term "placing", as used in claims 11 and 12.

Accordingly, withdrawal of the rejection is respectfully requested.

II. REJECTION OF CLAIMS 1-9 and 11-15 UNDER 35 USC §101

Claims 1-9 and 11-15 stand rejected under 35 USC §101 because the claimed invention allegedly is directed to non-statutory subject matter. The Examiner appears to focus on claim 1 and the fact that claim 1 does not expressly recite production of a simulation result.

Applicants respectfully disagree with the Examiner's rejection. Nevertheless, applicants have amended claim 1 to recite that the steps are performed so as to perform a simulation in order to obtain a simulation result. Applicants believe this amendment addresses the Examiner's concern with regard to claim 1.

Accordingly withdrawal of the rejection is respectfully requested.

III. REJECTION OF CLAIMS 1-7, 13-14 AND 16-19 UNDER 35 USC §102

Claims 1-7, 13-14 and 16-19 stand rejected under 35 USC §102(b) as being anticipated by U.S. Patent No. 5,870,588 to *Rompaey et al.* (hereinafter *Rompaey*). Withdrawal of the rejection is respectfully requested for at least the following reasons.

The Examiner asserts that the limitation in claim 1 "*wherein converting includes generating, for at least one discrete process, software code including a program loop having a jump instruction and a loop termination condition, and analyzing the loop termination condition to determine whether it is possibly non-terminating, and if so, replacing the jump instruction with an exit point*" does not recite how the generated loop is related to the surrounding steps or elements recited by the claim. The Examiner then states that "it appears to recite a sequence of <u>arbitrary steps</u> unrelated to the surrounding method", and "as a result, these limitations <u>do not further limit the claimed invention</u>" (emphasis added). Applicants respectfully disagree with the Examiner for at least the following reasons.

Claim 1 as originally filed <u>positively</u> recited (a) converting the at least one first model to at least one software model in the at least one first programming language. In the reply to the previous Office Action, claim 1 was amended to include the subject matter of claim 10, and now further <u>positively</u> recites (b) converting (i.e., part (a) above) <u>includes</u> generating, for at least one discrete process, software code including a program loop having a jump instruction and a loop termination condition, and analyzing the loop termination condition. Thus, claim 1, in its present form, requires both part (a) and part (b). Part (b) further defines part (a) and, thus, further limits the scope of the invention. It is respectfully submitted that the Examiner's statement to the contrary is incorrect.

With respect to the Examiner's comment that the additional step (i.e., part (b)) is arbitrary, it is respectfully noted that part (b) is described in the specification relative to part (a) (see, e.g., page 11, lines 20-24). Further, and as described in the specification, it is disclosed that

> "the sequential code generator 80 starts by analyzing the internal representation of the high-level model and builds an internal description of the sequential code of the process handler unit shown in figure 15" (see, e.g., page 44, lines 14-18 of the specification).

The high-level model is the "one first model", the intemal description of the sequential code of the process handler unit is the "one software model" represented in "one first programming language".

Furthermore, the specification discloses:

"The sequential code generator 80 builds the required sequential code by analysing the structure of the high-level model. The high-level model is based on a parallel algorithm and is therefore composed from sequential instructions by parallel composition and sequential constructs such as sequential composition and loops." (See, e.g., page 45, lines 8-14 of the specification).

Finally, it is disclosed that the sequential code generator:

(1) generates the sequential code for the termination condition from the high-level code for the same condition. It also generates a sequential code for the loop body from the high-level code of the body;

(2) analyzes the loop to check whether it is possibly non-terminating, or whether it definitely terminates;

(3) if a loop definitely terminates, then a similar loop is created by replacing the H-condition with the S-condition, and the H-body with the S-body; and

(4) if a loop may not terminate, then the beginning of the loop is marked with an entry point, and the jump instruction used for repeating the loop is replaced by an exit point and instruction to set the current process as handled. See, e.g., page 49, line 14-page 51, line 2 of the specification. Step (b) of pending claim one includes steps 1, 2 and 4.

In summary, applicants note that the one first model (i.e., the high-level model) is converted into the at least one software model (i.e., the intemal description of the sequential code of the process handler unit shown in figure 15) in the at least one first programming language, wherein the conversion step by the sequential code generator comprises:

i) generating, for a least one discrete process, software code (e.g., sequential code) including a program loop having a jump instruction in a loop termination condition;

ii) analyzing the loop termination condition to determine whether it is possibly non-terminating (or whether it definitely terminates); and

iii) if so, replacing the jump instruction with an exit point.

Thus, the interrelationships amongst the one first model, the software model, and the functions of the sequential code generator are clear from the disclosure in the specification. Further, and as discussed above, part (b), as recited in claim 1, further defines the operation of part (a), and thus narrows the scope of claim 1. In other words, part (b) above is not arbitrary, and part (b) further limits the scope of claim 1. Thus, part (a) and part (b) must be considered in determining patentablity of claim 1.

Rompaey has not been shown to teach these features of claim 1 and, therefore, *Rompaey* does not anticipate claim 1. Similar arguments apply to independent claims 16 and 19.

Accordingly, withdrawal of the rejection of claims 1, 16 and 19 is respectfully requested.

Claims 2-7, 13-14 and 17-18 depend from one of claims 1 or 16 and, therefore can be distinguished from *Rompaey* for at least the same reasons.

Accordingly, withdrawal of the rejection of claims 2-7, 13-14 and 17-18 is respectfully requested.

IV. REJECTION OF CLAIMS 8, 9, 11 AND 12 UNDER 35 USC §103

Claims 8, 9, 11 and 12 stand rejected under 35 USC §103(a) as being unpatentable over *Rompaey*. Claim 15 stands rejected under 35 USC §103(a) as being unpatentable over *Rompaey* in view of "Pentium® Pro Processor Performance Brief" (hereinafter "the brief"). Withdrawal of the rejection is respectfully requested for at least the following reasons.

As was noted above, claim 1 can be distinguished from *Rompaey*. Further, the brief has not been found to make up for the deficiencies of *Rompaey*. Thus, claim 1 can be distinguished over the combination of *Rompaey* and the brief. Since claims 8, 9, 11, 12 and 15 depend from claim 1, they can be distinguished from *Rompaey* and the brief for at least the same reasons.

Accordingly, withdrawal of the rejection of claims 8, 9, 11, 12 and 15 is respectfully requested.

Serial No. 10/014,831

V. *CONCLUSION*

Accordingly, claims 1-9 and 11-19 are believed to be allowable and the application is believed to be in condition for allowance. A prompt action to such end is earnestly solicited.

Should the Examiner feel that a telephone interview would be helpful to facilitate favorable prosecution of the above-identified application, the Examiner is invited to contact the undersigned at the telephone number provided below.

In the event any fee or additional fee is due in connection with the filing of this paper, the Commissioner is authorized to charge those fees to our Deposit Account No. 18-0988 (under the above Docket Number).

Respectfully submitted,

RENNER, OTTO, BOISSELLE & SKLAR, LLP

By

Mark D. Saralino, Reg. No. 34,243
Kenneth W. Fafrak, Reg. No. 50,689

1621 Euclid Avenue
Nineteenth Floor
Cleveland, Ohio 44115
PH: (216) 621-1113
FAX: (216)621-6165
B:\YAMA\P0795US\P0795 Reply_to_OA11-17-05.wpd